Computing and Informatics, Vol. 27, 2008, 453–465

REWRITING P SYSTEMS WITH CONDITIONAL COMMUNICATION: IMPROVED HIERARCHIES

H. Ramesh, Raghavan Rama

Department of Mathematics
Indian Institute of Technology, Madras
Chennai – 600 036, India
e-mail: {ramesh_h, ramar}@iitm.ac.in

Revised manuscript received 3 December 2007

Abstract. We consider here a variant of rewriting P systems [1], where communication is controlled by the contents of the strings, not by the evolution rules used for obtaining these strings. Some new characterizations of recursively enumerable languages are obtained by means of P systems with a small number of membranes, which improves some of the known results from [1] and [4].

Keywords: Membrane computing, rewriting P systems, recursively enumerable language

1 INTRODUCTION

P systems are a class of distributed parallel computing models inspired from the way the living cells process chemical compounds, energy, and information. Many variants of P systems use string objects and context-free rules for processing them. *Rewriting P systems* with string objects were introduced in [5]. Several variants of P systems with string objects have also been investigated extensively. In this work, we concentrate on rewriting P systems with conditional communication introduced in [1].

In this variant of rewriting P systems, the communication is controlled by the contents of the strings, not by the evolution rules themselves. This is achieved by considering certain types of *permitting* and *forbidding* conditions, based on the symbols or the substrings (arbitrary, or prefixes/suffixes) which appear in a given string.

Several characterizations of recursively enumerable languages were obtained in [1]. In [4], some of these results were improved. Here we give some new characterizations of recursively enumerable languages by means of P systems with a small number of membranes. These results improve some of the results from both [1] and [4]. In [1], there is a characterization of recursively enumerable language by systems, where both prefixes and suffixes are checked, without a bound on the number of membranes. It was also conjectured that the characterization holds also for a reduced number of membranes. We settle this here in an affirmative way by giving the characterization with 8 membranes.

2 SOME PREREQUISITES

In this section we introduce some formal language theory notions which will be used in this paper; for further details, we refer to [7].

For an alphabet V, we denote by V^* the set of all strings over V, including the empty one, denoted by λ. By RE we denote the family of recursively enumerable languages. The set of symbols appearing in a string x is denoted by $alph(x)$ and the substrings of x is denoted by $Sub(x)$.

In our proofs in the following sections we need the notion of a *matrix grammar with appearance checking*. Such a grammar is a construct $G = (N, T, S, M, F)$, where N, T are disjoint alphabets, $S \in N$, M is a finite set of sequences of the form $(A_1 \to x_1, \ldots, A_n \to x_n), n \geq 1$, of context-free rules over $N \cup T$, and F is a set of occurrences of rules in M (N is the nonterminal alphabet, T is the terminal alphabet, S is the axiom, while the elements of M are called matrices).

For $w, z \in (N \cup T)^*$ we write $w \Rightarrow z$ if there is a matrix $(A_1 \to x_1, \ldots, A_n \to x_n)$ in M and the strings $w_i \in (N \cup T)^*, 1 \leq i \leq n+1$, are such that $w = w_1, z = w_{n+1}$, and, for all $1 \leq i \leq n$, either (1) $w_i = w_i' A w_i''$, $w_{i+1} = w_i' x_i w_i''$, for some $w_i', w_i'' \in (N \cup T)^*$, or (2) $w_i = w_{i+1}$, A_i does not appear in w_i, and the rule $A_i \to x_i$ appears in F. (The rules of a matrix are applied in order, possibly skipping the rules in F if they cannot be applied – one says that these rules are applied in the *appearance checking mode*).

The language generated by G is defined by $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$. The family of languages of this form is denoted by MAT_{ac}. It is known that $MAT_{ac} = RE$.

A matrix grammar $G = (N, T, S, M, F)$ is said to be in the *binary normal form* if $N = N_1 \cup N_2 \cup \{S, \#\}$, with these three sets mutually disjoint, and the matrices in M are in the following forms:

1. $(S \to XA)$, with $X \in N_1, A \in N_2$,
2. $(X \to Y, A \to x)$, with $X, Y \in N_1, A \in N_2, x \in (N_2 \cup T)^*, |x| \leq 2$,
3. $(X \to Y, A \to \#)$, with $X, Y \in N_1, A \in N_2$,
4. $(X \to \lambda, A \to x)$, with $X \in N_1, A \in N_2$, and $x \in T^*, |x| \leq 2$.

Moreover, there is only one matrix of type 1 and F consists exactly of all rules $A \to \#$ appearing in matrices of type 3; $\#$ is a trap symbol – once introduced, it is never removed. A matrix of type 4 is used only once, in the last step of derivation.

According to [2], for each matrix grammar there is an equivalent matrix grammar in the binary normal form.

For an arbitrary matrix grammar $G = (N, T, S, M, F)$, let us denote by $ac(G)$ the cardinality of the set $\{A \in N \mid A \to \alpha \in F\}$. It was proved that each recursively enumerable language can be generated by a matrix grammar G such that $ac(G) \leq 2$. Consequently, to the properties of a grammar G in the binary normal form we can add the fact that $ac(G) \leq 2$. We will say that this is the *strong binary normal form* for matrix grammars.

There are several normal forms for type 0 grammars. We use the Penttonen normal form in our proofs. A type 0 grammar $G = (N, T, S, P)$ is said to be in *Penttonen normal form* if the rules from P are of one of the following forms: $A \to \lambda,\ A \to a,\ A \to BC,\ AB \to AC$, for $A, B, C \in N$ and $a \in T$.

3 REWRITING P SYSTEMS WITH CONDITIONAL COMMUNICATION

An extended *rewriting P systems* (of degree $m \geq 1$) *with conditional communication* is a construct

$$\Pi = (V, T, \mu, M_1, \ldots, M_m, R_1, P_1, F_1, \ldots, R_m, P_m, F_m),$$

where:

1. V is the alphabet;
2. $T \subseteq V$ is the terminal alphabet;
3. μ is the membrane structure;
4. $M_1, \ldots, M_m$ are finite languages over V, representing the strings intially present in the m regions;
5. $R_1, \ldots, R_m$ are finite sets of context-free rules over V present in the regions of μ;
6. P_i and F_i are permitting and forbidding conditions associated with the regions.

The conditions can be of the following forms:

empty: no restriction is imposed on strings, they either exit the current membrane or enter any of the directly inner membrane freely (but they cannot remain in the current membrane); we denote an empty permitting condition by $(true, X),\ X \in \{in, out\}$, and an empty forbidding condition by $(false, notX),\ X \in \{in, out\}$;

symbols checking: each P_i is a set of pairs $(a, X),\ X \in \{in, out\}$, for $a \in V$, and each F_i is a set of pairs $(b, notX),\ X \in \{in, out\}$, for $b \in V$; a string w can go

to a lower membrane only if there is a pair $(a, in) \in P_i$ with $a \in alph(w)$ and for each $(b, notin) \in F_i$ we have $b \notin alph(w)$; similarly for sending the string w out of membrane i it is necessary to have $a \in alph(w)$ for at least one pair $(a, out) \in P_i$ and $b \notin alph(w)$ for all $(b, notout) \in F_i$;

substring checking: each P_i is a set of pairs $(u, X),\ X \in \{in, out\}$, for $u \in V^+$, and each F_i is a set of pairs $(v, notX),\ X \in \{in, out\}$, for $v \in V^+$; a string w can go to a lower membrane only if there is a pair $(u, in) \in P_i$ with $u \in Sub(w)$, and for each $(v, notin) \in F_i$ we have $v \notin Sub(w)$; similarly for sending the string w out of membrane i it is necessary to have $u \in Sub(w)$ for at least one pair $(u, out) \in P_i$ and $v \notin Sub(w)$ for all $(v, notout) \in F_i$;

prefix/suffix checking: exactly as in the case of substrings checking, with the checked string being a prefix or a suffix of the string to be communicated.

We say that we have conditions of the types *empty, symb*, sub_k, $pref_k$, $suff_k$, where k is the length of the longest string in all P_i, F_i.

A system is said to be *non-extended* if $V = T$.

The transitions of the system are defined in the following way. In each region, each string which can be rewritten is rewritten by a rule from that region. The rule to be applied and the nonterminal it rewrites are non-deterministically chosen. The string obtained in this way is checked against the conditions $P_i,\ F_i$ from that region. If it fulfills the required conditions, then it will be immediately sent out of the membrane or to an inner membrane, if any exists; if it fulfills both *in* and *out* conditions, then it is sent to a membrane non-deterministically chosing the direction – and non-deterministically choosing the inner membrane in the case when several directly inner membranes exist. If a string does not fulfill any condition, or it fulfills only *in* conditions and there is no inner membrane, then the string remains in the same region. If a string cannot be rewritten, then it is directly checked against the communication conditions. That is, the rewriting has priority over communication.

A sequence of transitions form a computation and the result of a halting computation is the set of strings over T sent out of the system. In the case of non-extended systems, all strings sent out are accepted. A computation does not yield a result if it does not halt. A string which remains inside the system or, in the case of extended systems, which exits but contains nonterminal symbols does not contribute to the generated language. The language generated by a system Π is denoted by $L(\Pi)$.

We denote by $RP_n(rw, \alpha, \beta)$, $n \geq 1$, $\alpha, \beta \in \{empty, symb\} \cup \{sub_k \mid k \geq 2\} \cup \{prefsuff_k \mid k \geq 2\}$, the family of languages generated by P system of degree at most n and with permitting and forbidding conditions of type α and β, respectively.

4 IMPROVED UNIVERSALITY RESULTS

In [1], it was proved that P systems of degree 4 with permitting conditions of type sub_2 and forbidding conditions of type *symb* are computationally universal. This

result has been improved from 4 to 3 membranes in [4]. We improve this result and show that universality can be achieved with 2 membranes in this case.

Theorem 1. $RE = RP_2(rw, sub_2, symb)$.

Proof. Let us consider a type 0 grammar $G = (N, T, S, P)$, in Penttonen normal form, with the non context-free rules from P labelled in a one-to-one manner and construct the system

$$\Pi = (V, T, [_1[_2]_2]_1, \{S\}, \emptyset, (R_1, P_1, F_1), (R_2, P_2, F_2)),$$

with the following components:

$$\begin{aligned}
V &= N \cup T \cup \{(B, r) \mid r : AB \to AC \in P\};\\
R_1 &= \{A \to x \mid A \to x \in P\} \cup\\
&\quad \{B \to (B, r) \mid r : AB \to AC \in P\};\\
P_1 &= \{(A(B, r), in) \mid r : AB \to AC \in P\} \cup\\
&\quad \{(true, out)\};\\
F_1 &= \{(false, notout)\};\\
R_2 &= \{(B, r) \to C \mid r : AB \to AC \in P\};\\
P_2 &= \{(true, out)\};\\
F_2 &= \{((B, r), notout) \mid r : AB \to AC \in P\}.
\end{aligned}$$

The system works as follows: The initial configuration of the system is $[_1S[_2]_2]_1$. The context-free rules from P are present in R_1 as rewriting rules, hence we can simulate them without any difficulty. Let us assume that we have a string w_1ABw_2 in membrane 1. In order to simulate a rule $r : AB \to AC \in P$, we apply the rule $B \to (B, r)$ on the string. The string is sent to membrane 2 only if it has a substring of the form $A(B, r)$ for some $r : AB \to AC \in P$. Otherwise, the string is sent out, but it is not a terminal one. In membrane 2, we replace the symbol (B, r) with C and send the resulting string to the skin membrane. In this way, we complete the simulation of the non context-free rule.

The process can be iterated until no nonterminal is present in the sentential form. Hence, each derivation in G can be simulated in Π and, conversely, all halting computations in Π correspond to correct derivations in G. Therefore, the computation in Π can stop only after reaching a terminal string with respect to G. Thus, we have $L(G) = L(\Pi)$. □

In [1], it was proved that P systems of degree 4 with permitting conditions of type sub_2 and forbidding conditions of type *empty* can characterize recusively enumerable languages. We improve this result by proving the universality with 3 membranes.

Theorem 2. $RE = RP_3(rw, sub_2, empty)$.

Proof. We start again from a type 0 grammar $G = (N, T, S, P)$ in Penttonen normal form, with the non context-free rules in P labelled in a one-to-one manner, and we construct the P system

$$\Pi = (V, T, [_1[_2]_2[_3]_3]_1, \emptyset, \{S\}, \emptyset, (R_1, P_1, F_1), \ldots, (R_3, P_3, F_3)),$$

with the following components:

$$\begin{aligned}
V &= N \cup T \cup \{(B, r) \mid r : AB \to AC \in P\} \cup \\
&\quad \{A', A'' \mid A \in N\} \cup \{f, Z\}; \\
R_1 &= \{f \to \lambda, C'' \to Z\} \cup \\
&\quad \{C' \to C'' \mid C \in N\}; \\
P_1 &= \{(\lambda, out)\} \cup \{(C'', in) \mid C \in N\} \cup \\
&\quad \{(A(B, r), in) \mid r : AB \to AC \in P\}; \\
R_2 &= \{B \to (B, r) \mid r : AB \to AC \in P\} \cup \\
&\quad \{A \to x,\ A \to xf \mid A \to x \in P\} \cup \\
&\quad \{C'' \to C \mid C \in N\}; \\
P_2 &= \{(f, out)\} \cup \\
&\quad \{((B, r), out) \mid r : AB \to AC \in P\}; \\
R_3 &= \{(B, r) \to C' \mid r : AB \to AC \in P\} \cup \\
&\quad \{C'' \to Z \mid C \in N\}; \\
P_3 &= \{(C', out) \mid C \in N\}.
\end{aligned}$$

All sets of forbidding conditions consist of the pairs $(false, notin)$, $(false, notout)$.

This system works as follows. We start in membrane 2 with the axiom of G. The context-free rules of G can be simulated here. If a terminal rule $A \to xf$ is used in membrane 2, then the string goes to membrane 1 and from here out of the system. If it is not terminal, it is not accepted in the generated language. If the string is terminal, then it is introduced in $L(\Pi)$.

Suppose that a string w is rewritten in membrane 2 by a rule $B \to (B, r)$ associated with a rule $r : AB \to AC \in P$. It exits; if the symbols A and (B, r) are not associated with the same rule from P, then the string is sent out, but it is not a terminal one. Assume that the string is of the form $w_1 A(B, r) w_2$, for some $r : AB \to AC \in P$. No rule can be applied in membrane 1, but the string can be sent to a lower membrane. If it arrives back in membrane 2, then it will exit either unchanged or after introducing one more symbol of the form (B, r). The process is repeated; eventually, the string will arrive in membrane 3 (otherwise we either continue between membrane 1 and 2 or we send the string out of the system and it is not a terminal one). Here in membrane 3 we replace the symbol (B, r) with C' and the string is sent back to the skin membrane. In the skin membrane, the symbol C' is replaced with C''.

Now there are two cases. If we had at least two symbols of the form (B, r) and (B_1, r_1) in the string, then before finishing the simulation of the rule r, we can start the simulation of the rule r_1; but then the trap symbol Z will be introduced. So we have to finish the simulation of the rule r first. In the other case, the string can be sent to one of membranes 2 and 3. If the string arrives back to 3, then the trap symbol will be introduced. Thus, we have to send the string to membrane 2. We have two cases here. If in membrane 2 we use the rule $C'' \to C$, then we have again a string $(N \cup T)^*$, and the process can be iterated. If before using the rule $C'' \to C$, we use a rule $B \to (B, r)$, then the string should go to membrane 1 where we introduce the trap symbol Z by the rule $C'' \to Z$. Thus $L(G) = L(\Pi)$. □

The universality result for P systems with both permitting and forbidding conditions of type *symb* has been improved from 6 [1] to 5 membranes in [4]. Here we give a universality result with only 3 membranes. We use the same idea as in [3].

Theorem 3. $RE = RP_3(rw, symb, symb)$.

Proof. Consider a matrix grammar with appearance checking $G = (N, T, S, M, F)$ in the strong binary normal form with $N = N_1 \cup N_2 \cup \{S, \#\}$. Assume that $ac(G) = 2$, and let $B^{(1)}$ and $B^{(2)}$ be the two objects in N_2 for which we have rules $B^{(j)} \to \#$ in matrices of M. Let us assume that we have k matrices of the form $m_i : (X \to \alpha, A \to x), X \in N_1, \alpha \in N_1 \cup \{\lambda\}, A \in N_2$, and $x \in (N_2 \cup T)^*$. We replace each matrix of the form $(X \to \lambda, A \to x)$ by $(X \to f, A \to x)$ where f is a new symbol. We continue to lable the obtained matrices in the same way as the original one. The matrices of the form $(X \to Y, B^{(j)} \to \#)$, are labeled by m_i with $i \in lab_j$, for $j = 1, 2$ such that lab_1, lab_2 and $lab_0 = \{1, 2, \ldots, k\}$ are mutually disjoint sets.

We construct the P system (of degree 3)

$$\Pi = (V, T, \mu, M_1, \ldots, M_3, R_1, P_1, F_1, \ldots, R_3, P_3, F_3),$$

with the following components:

$$\begin{aligned}
V &= N_1 \cup N_2 \cup T \cup \{X_{i,j} \mid X \in N_1, 1 \le i \le k, 0 \le j \le k\} \cup \\
&\quad \{A_i, A_{i,j} \mid A \in N_2, 1 \le i \le k, 0 \le j \le k\} \cup \\
&\quad \{X', X'', X^{(1)}, X^{(2)} \mid X \in N_1 \cup \{f\}\}, \\
\mu &= [_1[_2[_3]_3]_2]_1, \\
M_1 &= \{XA\}, \text{ for } (S \to XA) \text{ being the initial matrix of } G, \\
M_2 &= M_3 = \emptyset,
\end{aligned}$$

and with the following triples $(R_i, P_i, F_i), 1 \le i \le 3$:

$$\begin{aligned}
R_1 &= \{X \to Y^{(1)} \mid m_i : (X \to Y, B^{(1)} \to \#\} \cup \\
&\quad \{X \to Y^{(2)} \mid m_i : (X \to Y, B^{(2)} \to \#\} \cup \\
&\quad \{A \to A_{i,0} \mid m_i : (X \to \alpha, A \to x), 1 \le i \le k, \alpha \in N_1 \cup \{f\}\} \cup
\end{aligned}$$

$$
\begin{aligned}
& \{A_{i,j} \to \# \mid 1 \le j < i \le k\} \cup \\
& \{A_i \to x \mid m_i : (X \to \alpha, A \to x), 1 \le i \le k, \alpha \in N_1 \cup \{f\}\} \cup \\
& \{\alpha' \to \alpha \mid \alpha \in N_1\} \cup \{f' \to \lambda\}; \\
P_1 &= \{(A_{i,0}, in) \mid i \le i \le k\} = \\
& \{(X^{(1)}, in), (X^{(2)}, in) \mid X \in N_1\} = \\
& \{(a, out) \mid a \in T\}; \\
F_1 &= \{(X, notout) \mid X \in N_1 \cup N_2\} \cup \\
& \{(A_i, notin) \mid A \in N_2, 1 \le i \le k\} \cup \\
& \{(A_{i,j}, notout) \mid A \in N_2, 1 \le i, j \le k\} \cup \\
& \{(\alpha', notin) \mid \alpha \in N_1 \cup \{f\}\}; \\
R_2 &= \{X \to X_{i,0} \mid m_i : (X \to \alpha, A \to x), 1 \le i \le k, \alpha \in N_1 \cup \{f\}\} \cup \\
& \{B^{(1)} \to \#,\ \# \to \#\} \cup \\
& \{Y^{(1)} \to Y,\ Y'' \to Y \mid Y \in N_1\} \cup \\
& \{X_{i,j} \to X_{i,j+1} \mid X \in N_1, 1 \le j < i \le k\} \cup \\
& \{X_{i,i} \to \alpha' \mid m_i : (X \to \alpha, A \to x), 1 \le i \le k, \alpha \in N_1 \cup \{f\}\}; \\
P_2 &= \{(Y, out), (Y^{(2)}, in) \mid Y \in N_1\} \cup \\
& \{(X_{i,j}, in) \mid 1 \le j < i \le k\} \cup \\
& \{(\alpha', out) \mid \alpha \in N_1 \cup \{f\}\}; \\
F_2 &= \{(B^{(1)}, notout) \mid B \in N_2\} \cup \\
& \{(\alpha'', notin) \mid \alpha \in N_1 \cup \{f\}\}; \\
R_3 &= \{Y^{(2)} \to Y''\} \cup \{B^{(2)} \to \#, \# \to \#\} \cup \\
& \{A_{i,j} \to A_{i,j+1} \mid A \in N_2, 1 \le j < i \le k\} \cup \\
& \{A_{i,i} \to A_i,\ A_i \to \# \mid 1 \le i \le k\}; \\
P_3 &= \{(Y', out) \mid Y \in N_1\} \cup \\
& \{(A_{i,j}, out) \mid 1 \le j < i \le k\} \cup \\
& \{(A_i, out) \mid 1 \le i \le k\}; \\
F_3 &= \{(B^{(2)}, notout) \mid B \in N_2\}.
\end{aligned}
$$

Only strings over T are accepted in the generated language; # is a trap symbol. From the skin membrane in any moment we can send out a string if it contains at least one terminal symbol, but the string is not accepted in the generated language if it contains any symbol not in T.

Simulation of the matrix $m_i : (X \to \alpha, A \to x), 1 \le i \le k$**:** We start the simulation by the rule $A \to A_{i,0}$. The string can be sent to membrane 2 where we apply the rule $X \to X_{j,0}$. The obtained string is sent to membrane 3. From now on, the string will go back and forth between membranes 2 and 3, and the second subscript of the symbols $X_{i,s}$ and $Y_{j,t}$ is alternatively increased. Now we have three cases here:

Case 1: $i < j$**.** This means that at some step in membrane 3 we have a string of the form $X_{j,i}w_1A_{i,i-1}w_2$. We replace $A_{i,i-1}$ with $A_{i,i}$ and no communication is possible. So we use the rule $A_{i,i} \to A_i$ and the string is sent out. In membrane 2, we replace $X_{j,i}$ with $X_{j,i+1}$ and send the string back to membrane 3, where the trap symbol # is introduced (the rewriting has priority over communication).

Case 2: $i > j$**.** At some moment we have a string of the form $X_{j,j}w_1A_{i,j-1}w_2$ in membrane 2 which is sent to membrane 3. Here we replace $A_{i,j-1}$ with $A_{i,j}$ and send the string out. In membrane 2 we replace $X_{j,j}$ with α', and the string is sent out. In the skin membrane, we can apply $A_{i,j} \to \#$, hence the string will never lead to a terminal one.

Case 3: $i = j$**.** At some moment we pass from membrane 2 to membrane 3 a string $X_{i,i}w_1A_{i,i-1}w_2$. In membrane 3, we replace $A_{i,i-1}$ with $A_{i,i}$ and, because we cannot exit, we replace $A_{i,i}$ with A_i and send out the string. Here in membrane 2 we replace $X_{i,i}$ with α' and send the string to the skin membrane. In the skin membrane we have to replace α' with α and A_i with x before starting the simulation of the next matrix.

Simulation of the matices $(X \to Y, B^{(j)} \to \#), j = 1, 2$**:** The simulation of a matrix of this form starts by a rule $X \to Y^{(1)}$ or $X \to Y^{(2)}$ in the skin membrane. If we are simulating a rule $(X \to Y, B^{(1)} \to \#)$, then in membrane 2 we use the rule $Y^{(1)} \to Y$. Now the string can be sent to the skin membrane only if $B^{(1)}$ is not present. Similarly, if we are simulating $(X \to Y, B^{(2)} \to \#)$, then in membrane 2 there is no rule we can apply. So we send the string to membrane 3, where we replace $Y^{(2)}$ with Y'' and the resulting string can be sent out if $B^{(2)}$ is not present. Back in membrane 2, we replace Y'' with Y and send the string to the skin membrane.

If at any moment we get a string of the form $f'w$, for $w \in T^*$, in the skin membrane, then we remove f' and send the string out. Consequently, $L(G) = L(\Pi)$. □

There is a characterization of recursively enumerable languages by P systems with permitting conditions of type $prefsuff_2$ and forbidding conditions of type *empty* in [1] without a bound on the number of membranes. It was conjectured that such a characterization holds also for a reduced number of membranes. We settle this conjecture in the positive here and show that *eight* membranes are enough for achieving the universality.

Theorem 4. $RE = RP_8(rw, prefsuff_2, empty)$.

Proof. Let us consider a type 0 grammar $G = (N, T, S, P)$ in Penttonen normal form, with the non context-free rules in P labelled in an injective manner, and assume that $N \cup T \cup \{\$\} = \{E_1, E_2, \ldots, E_n\}$. We construct the P system Π, of degree 8, with the following components:

$$
\begin{aligned}
V &= N \cup T \cup \{A' \mid A \in N\} \cup \\
&\quad \{X, Y, Y', Z, \$\} \cup \\
&\quad \{X_i, Y_i, X_{i,j}, Y_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq n\} \cup \\
&\quad \{(B, r) \mid r : AB \to AC \in P\}, \\
\mu &= [_1[_2[_3[_4[_5[_6]_6]_5]_4]_3[_7[_8]_8]_7]_2]_1, \\
M_i &= \emptyset, 1 \leq i \leq 8, i \neq 2, \\
M_2 &= X\$SY,
\end{aligned}
$$

and with the following sets of rules and associated permitting conditions. All forbidding condition sets are of the form $\{(false, notin), (false, notout)\}$:

$$
\begin{aligned}
R_1 &= \{X \to \lambda,\ Y \to \lambda,\ \$ \to \lambda\}; \\
P_1 &= \{(a, out) \mid a \in T\}; \\
R_2 &= \{E_i \to E_i',\ Y \to Y_{i,0} \mid 1 \leq i \leq n\} \cup \\
&\quad \{B \to (B, r) \mid r : AB \to AC \in P\} \cup \\
&\quad \{A \to x \mid A \to x \in P\} \cup \\
&\quad \{(B, r) \to Z \mid r : AB \to AC \in P\} \cup \\
&\quad \{E_i' \to Z,\ Y_{i,0} \to Z \mid 1 \leq i \leq n\}; \\
P_2 &= \{(\$Y, out)\} \cup \\
&\quad \{(E_i'Y_{i,0}, in) \mid r : AB \to AC \in P,\ 1 \leq i \leq n\} \cup \\
&\quad \{((B, r)Y, in) \mid r : AB \to AC \in P\}; \cup \\
R_3 &= \{E_i' \to \lambda, Y_i \to Y \mid 1 \leq i \leq n\} \cup \\
&\quad \{(B, r) \to Z \mid r : AB \to AC \in P\}; \\
P_3 &= \{(Y_{i,0}, in) \mid 1 \leq i \leq n\} \cup \\
&\quad \{(Y, out)\}; \\
R_4 &= \{X \to X_{i,0}E_i,\ X_i \to X \mid 1 \leq i \leq n\} \cup \\
&\quad \{Y_{i,j} \to Z \mid 1 \leq j < i \leq n\}; \\
P_4 &= \{(X_{i,0}, in) \mid 1 \leq i \leq n\} \cup \\
&\quad \{(X, out)\}; \\
R_5 &= \{X_{i,j} \to X_{i,j+1} \mid 0 \leq j < i \leq n\} \cup \\
&\quad \{X_{i,i} \to X_i \mid 1 \leq i \leq n\}; \\
P_5 &= \{(X_{i,j}, in) \mid 1 \leq j < i \leq n\} \cup \\
&\quad \{(X_i, out) \mid 1 \leq i \leq n\}; \\
R_6 &= \{Y_{i,j} \to Y_{i,j+1} \mid 0 \leq j < i \leq n\} \cup \\
&\quad \{Y_{i,i} \to Y_i,\ Y_i \to Z \mid 1 \leq i \leq n\}; \\
P_6 &= \{(Y_{i,j}, out) \mid 1 \leq j < i \leq n\} \cup \\
&\quad \{(Y_i, out) \mid 1 \leq i \leq n\};
\end{aligned}
$$

$$
\begin{aligned}
R_7 &= \{Y \to \lambda,\ Y' \to Y\} \cup \\
&\quad \{Y_{i,0} \to Z \mid 1 \le i \le n\}; \\
P_7 &= \{(A(B,r), in),\ (CY, out) \mid r : AB \to AC \in P\}; \\
R_8 &= \{(B,r) \to CY' \mid r : AB \to AC \in P\}; \\
P_8 &= \{(CY', out)\}.
\end{aligned}
$$

We start from the string $X\$SY$, intially present in membrane 2. We plan to simulate the non context-free rules from P in the right end ofthe strings of Π and to this aim we use the so-called *rotate-and-simulate* technique much used in the DNA computing area. If $Xw_1\$w_2Y$ is a sentential form of Π, then w_2w_1 is a sentential form of G. The symbol \$ indicates the actual beginning of strings from G. Z is a trap symbol, once introduced, it cannot be removed, hence the string will never become a terminal one.

In membrane 2, we can simulate any context-free rule from P and the string will remain in the same region. We start the procedure of circularly permuting the string with one symbol by using the rules $E_i \to E_i'$ and $Y \to Y_{i,0}$. If the primed symbol is the rightmost one, then the condition to send the string to a lower membrane is fulfilled. If we did not use both the rules or the primed symbol is not the rightmost one, then the trap symbol is introduced. Now we can either send the string to membrane 7 or 3. If it enters membrane 7, then we introduce the trap symbol.

In membrane 3, we remove E_i' and send the string to membrane 4. In membrane 4, we replace X with $X_{i,0}E_i$ and send the string to membrane 5. From now on, the string will go back and forth between membranes 5 and 6, and the second subscript of the symbols $X_{i,s}$ and $Y_{j,t}$ is alternatively increased. Now there are three cases:

Case 1: $i < j$**.** This means that at some step in membrane 6 we have a string $X_{j,i}wY_{i,i-1}$. We replace $Y_{i,i-1}$ with $Y_{i,i}$ and no communication is possible, hence one more rewriting is necessary. We replace $Y_{i,i}$ with Y_i and the string is sent out. In membrane 5 we replace $X_{j,i}$ with $X_{j,i+1}$ and the string is sent back to membrane 6, where we introduce the trap symbol.

Case 2: $i > j$**.** At some moment we have a string of the form $X_{j,j}wY_{i,j-1}$ in membrane 5, which is sent to membrane 6. We replace $Y_{i,j-1}$ with $Y_{i,j}$ in membrane 6 and the string exits. In membrane 5 we replace $X_{j,j}$ with X_j and the string is sent out. Back in membrane 4 we can apply $Y_{i,j} \to Z$, hence the string will never lead to a terminal one.

Case 3: $i = j$**.** At some moment we pass from membrane 5 to membrane 6 a string $X_{i,i}wY_{i,i-1}$. In membrane 6 we replace $Y_{i,i-1}$ with $Y_{i,i}$ and, because we cannot exit, we replace $Y_{i,i}$ with Y_i and sent the string out. In membrane 5 we replace $X_{i,i}$ with X_i and the string is sent out. We replace the symbols X_i and Y_i with X and Y, respectively in membrane 4 and 5 and the string is sent to membrane 2. The process of circularly permuting the sym-

bol will end successful if we add the symbol E_i in the left end of the string corresponding to the symbol E_i' which was removed from the right end of the string

We simulate the non context-free rules $r : AB \to AC$ in the following way. A symbol B is replaced by (B, r) in membrane 2, if this is not done in the rightmost position, then the symbol Z is introduced. If the string is of the form $Xw(B, r)Y$, then it has to go to membrane 7. In membrane 7 we replace the symbol Y and send the string to membrane 8, if the string is of the form $Xw_1A(B, r)$ corresponding to some rule $r : AB \to AC \in P$. In membrane 8 we replace (B, r) with CY' and send out the resulting string. In membrane 7 we replace Y' with Y and send the string to membrane 2.

The process can be iterated. Consequently, $L(G) = L(\Pi)$. □

5 CONCLUSION

In this paper we gave some improved results about rewriting P systems with conditional communication. We believe that the result of Theorem 3 cannot be improved further. It is an open problem whether or not the result of Theorem 4 can be improved.

REFERENCES

[1] Bottoni, P.—Labella, A.—Martin Vide, C.—Păun, Gh.: Rewriting P Systems with Conditional Communication. LNCS, Springer, Vol. 2300, 2002, pp. 325–353.

[2] Dassow, J.—Păun, Gh.: Regulated Rewriting in Formal Language Theory. Springer-Verlag, 1989.

[3] Krishna, S. N.—Rama, R.—Ramesh, H.: Further Results on Contextual and Rewriting P Systems. Fundamenta Informaticae, Vol. 64, 2005, No. 1–4, pp. 241–253.

[4] Madhu, M.: Rewriting P Systems: Improved Hierarchies. Theoretical Computer Science, Vol. 334, 2005, pp. 161–175.

[5] Păun, Gh.: Computing with Membranes. Journal of Computer and System Sciences, Vol. 61, 2000, No. 1, pp. 108–143.

[6] Păun, Gh.: Membrane Computing: An Introduction. Springer-Verlag, Berlin, 2002.

[7] Rozenberg, G.—Salomaa, A. (Eds.): Handbook of Formal Languages (3 Volumes). Springer, 1997.

H. Ramesh is a Ph. D. student in the Department of Mathematics at Indian Institute of Technology in Madras, India. He is currently working with Prof. R. Rama. His research interests are formal languages and automata theory, DNA computing and membrane computing.

Raghavan Rama is professor in the Department of Mathematics at Indian Institute of Technology in Madras, India. Her main research fields are formal language theory, automata theory, DNA computing and membrane computing. She has (co)authored more than 40 research papers in conference proceedings and journals such as Theoretical Computer Science, International Journal of Computer Mathematics, Journal of Automata Languages and Combinatorics, International Journal of Pattern Recognition and Artificial Intelligence, Fundamenta Informaticae.